

The Integrated Workstation,
A Realtime Data Acquisition, Analysis and Display System

Thomas R. Treadway III

This paper was prepared for submittal to
1991 INTEREX HP Users Conference
San Diego, CA August 5-8, 1991

May 1991

Lawrence
Livermore
National
Laboratory

This is a preprint of a paper intended for publication in a journal or proceedings. Since changes may be made before publication, this preprint is made available with the understanding that it will not be cited or reproduced without the permission of the author.

**CIRCULATION COPY
SUBJECT TO RECALL
IN TWO WEEKS**

DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

The Integrated Workstation, A Realtime Data Acquisition, Analysis and Display System

Thomas R. Treadway III

Lawrence Livermore National Laboratory
P.O. Box 808 L-573
Livermore, CA 94550
(415) 423-4997
treadway@llnl.GOV

Abstract

The Integrated Workstation was developed at Lawrence Livermore National Laboratory to consolidate the data from many widely dispersed systems in order to provide an overall indication of the enrichment performance of the Atomic Vapor Laser Isotope Separation experiments. In order to accomplish this task a Hewlett Packard 9000/835 turboSRX was employed to acquire over 150 analog input signals. Following the data acquisition, a spreadsheet-type analysis package and interpreter was used to derive 300 additional values. These values were the results of applying physics models to the raw data. Following the calculations data were plotted and archived for post-run analysis and report generation. Both the modeling calculations, and real-time plot configurations can be dynamically reconfigured as needed. Typical sustained data acquisition and display rates of the system was 1 Hz. However rates exceeding 2.5 Hz have been obtained. This paper will discuss the instrumentation, architecture, implementation, usage, and results of this system in a set of experiments that occurred in 1989.

Introduction

Lawrence Livermore National Laboratory (LLNL) is developing the Atomic Vapor Laser Isotope Separation (AVLIS) technology. This technology uses the world's highest average power visible light lasers to enrich natural uranium for use as fuel for commercial light water nuclear reactors. Development of AVLIS will provide the United States with the most cost effective enrichment technology available. The AVLIS process involves the vaporizing of uranium with an electron beam. These vaporized atoms are illuminated by laser beams having very precisely tuned optical frequencies. The laser beams photoionize (i.e. remove an electron from) the U-235 isotope, giving it a positive electrical charge, while leaving the other isotopes neutrally charged. These ionized atoms are attracted to a negatively charged surface, while the un-ionized U-238 atoms pass through the ion extraction zone. This technique is also applicable to many other elements.

An Engineering Demonstration System using the AVLIS technology was developed and tested in 1989 at LLNL. This facility required a computer system to consolidate the diagnostic data and to provide a real-time indication of the overall enrichment and photoionization performance of the process. The system dedicated to this task was called the Integrated Workstation (IWS). Accomplishment of this task required the acquisition of over 150 analog signals, along with the calculation (derivation) of approximately 300 additional values. Analysis was performed using the above input signals along with results from intermediate calculations, as input data to physical models. Following the analysis, the data were plotted and archived. The IWS provides the physicist and experimenters access to all of this data upon a 19-inch, high resolution color graphics display, in the form of real-time trend plots.

All of the data was available for nearly any mathematical calculation desired during the experiments, and more extensive interactive post-run analysis. The IWS's default configuration allows for up to 64 plots, each plot displaying up to 8 variables on the same axes simultaneously, for a total of 256 individual variables displayed simultaneously. Designed with maximum flexibility in mind, the plot configuration and modeling calculation executed during real-time could be changed dynamically.

History

Early AVLIS experiments were performed using the Hewlett Packard (HP) 1000 model A900 computer, using two 32 channel data loggers each coupled with a 64K memory module, installed on a CAMAC crate. Data scans were typically taken over a 20 to 50 second period, followed by down loading of the results from the CAMAC data logger to computer memory over HP-IB (IEEE 488). Once the data was in computer memory and a copy written to disk, another scan would be triggered, and the process repeated. While the data loggers were sampling at 20 to 50 Hz, the computer would be averaging the raw data of the previous scan. Following the averaging, the 12-bit integer data were converted to volts, or their respective engineering units. Subsequent to the data acquisition, the modeling calculations were performed by a powerful, real-time modifiable spreadsheet code, followed by the plotting. User selected data was plotted upon a graphics display, which was tightly coupled with a button box, allowing for the instant selection of any one of 8 multivariate sets of plots.

The entire sequence from the CAMAC data read to the final plot was typically done in just under 20 seconds (at least a half dozen other applications were running on this HP at the same time). Post run this same HP 1000 was used for local data analysis and generation of tapes for analysis by the Cray. Similar characteristics were needed from the Integrated Workstation in support of the AVLIS demonstration, but at least 4 times better performance was desired.

The IWS started as a software port of the HP 1000 system, but this time on a new computer; an HP 9000 model 825 with an SRX graphics subsystem. Later it was upgraded to a model 835 with turboSRX. Porting of the software involved modification, testing, and debugging of about 80,000 lines of FORTRAN code. The initial (unoptimized) post-migration test provided only a 2 times execution speed improvement compared to the HP 1000 version. Unfortunately, this new implementation would require that the IWS support at least 4 remotely located data acquisition units, and about 4 times more data.

Hardware Architecture

HP 3852A Data Acquisition / Control Units (scanners) were substituted for the CAMAC front ends, which yielded very impressive results. The HP 3852's were no more expensive than the CAMAC solution, but proved to be much more reliable and flexible.

Each scanner functioned like a standalone computer with its own real-time multi-tasking Basic programming language that supported data acquisition and control. The back of the scanner consisted of an eight slot card-cage that accepted a multitude of different modules such as: high-speed voltmeters, high-speed multiplexers (that read VDC, VAC, thermocouples, current, ohms, and strain-gages), analog output, stepper motor control, pulse counters, digital input/output, breadboard, and even a HP-IB disk or tape drive interface controller.

Instrumentation

Each scanner used a HP 44702B 13-bit High-Speed Digital Voltmeter (100,000 reading/second), which took up two slots, as the primary acquisition unit. By adding multiple HP 44711A 24-Channel High-Speed FET Multiplexers, almost 150 signals could be acquired by a single scanner before an expansion chassis was needed. One expansion chassis would increase this number to over 350 signal channels, and the maximum of seven extender chassis would allow over 1500 channels.

The flexibility of the HP 3852's facilitated the straight forward addition of special input signal measurements by the plugging in of the appropriate data acquisition module into the scanner. Two other modules were used: an HP 44713A 24-Channel High-Speed FET Multiplexer with Thermocouple Compensation for measuring temperatures and an HP 44715A 5-Channel Counter/Totalizer module for pulse counting up to 200 KHz.

The data acquisition configuration had 4 scanners trigger-synchronized to each other. One scanner was used to control the initiation of a scan, typically once every second. While sampling was at 600 Hz, an averaging was done for each channel to remove any 60 Hz line noise. The data was then read by the computer from each scanner as double-precision floating-point voltages, temperature, or counts depending on the signal. The HP 3852A's are HP-IB instruments, so in order to span the distance to the computer, HP 37204A HP-IB Extenders were used. The preprocessing of the data on the front ends, reduced the amount of work the computer needed to do, as well as reduced the amount of data transferred. With the data in computer memory, each point was converted to engineering units. All of this was done in well under a second. The IWS hardware configuration is shown in Figure 1.

Graphics System

Selected data signals along with results from analysis calculations were displayed on a high resolution 19 inch color graphics display monitor. A 32 button box, keyboard, and mouse, connected by HP-IL were the input devices for the user of the IWS.

Graphics was provided by the HP 98731A turboSRX graphics subsystem. The monitor displayed the user selected plots, upon the 24-bit graphics planes at 1280x1024 pixel resolution. Superimposed upon the graphics planes was an additional 4-bit overlay planes in which X-windows was running. When an opaque window was displayed in the overlay planes, the image in the graphics planes would be blocked (obscured). The 24-bit graphics planes of the turboSRX are composed of three 8-bit banks. One bank was used by each of the primary colors: red, green and blue. The intensity for each bit in every bank was set to maximum intensity. By accessing only 1-bit at a time out of each bank, eight 3-bit graphics planes were created. A 3-bit graphics plane will allow up to 8 colors to be defined; 3 primary, 3 secondary, black and white.

The 32 function buttons allowed the user to switch between the any one of the eight 3-bit graphics planes containing the plots, zoom in or out (thanks to the turboSRX), as well as pan across the screen. The keyboard was used for command input by the user, and the 3 button mouse used for X-windows operations. A typical experimental configuration had a terminal emulation window in the lower left hand corner of the overlay screen, through which the user entered commands to the IWS.

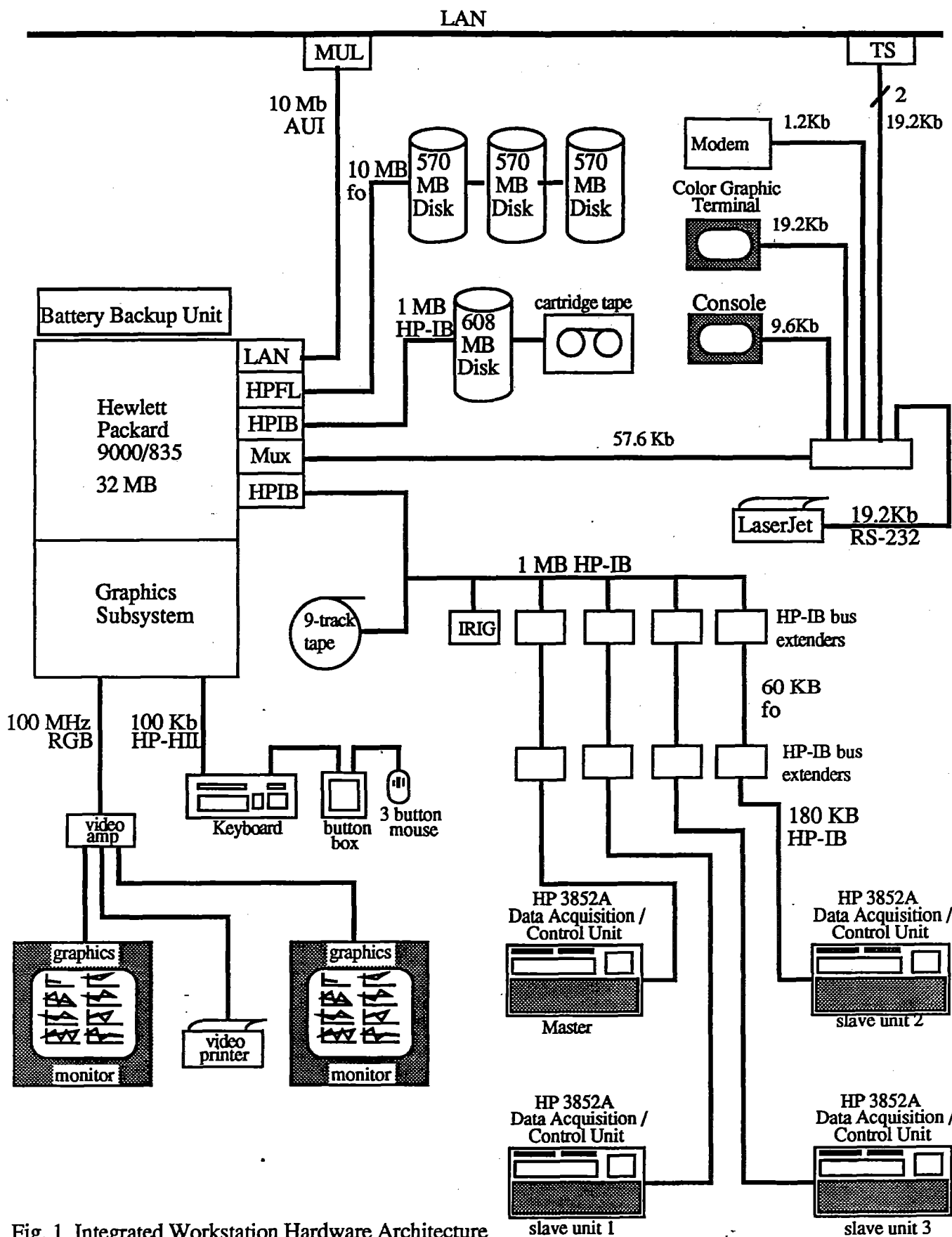


Fig. 1 Integrated Workstation Hardware Architecture

Software Architecture

The IWS contained a set of flexible independent software programs. Flexibility and speed were the two most important considerations driving the design of the IWS. In experimental environments in which the IWS operated, many changes must be accommodated, including varying the data signals, acquisition rates, analysis calculations performed, and the experimental objectives. Most of the internal data structure was table driven, thereby reducing the amount of code that needed to be re-compiled (in most cases to none). Also, through the use of interpretive interfaces the system possessed a high level of dynamic reconfigurability. What is meant by "independent" is that each program had a specific task to perform in the sequence of the data processing, yet could be removed (disabled) from the processing pipeline without affecting the operation of the overall system.

User Interface

Most visible of all the IWS programs was the User Interface (**ui**) program. From this program the user communicated operations and commands to the IWS. The **ui** program was an interactive command line interpreter, which read its commands from the keyboard or optionally from macro files. All of the interactive IWS programs allowed an option that would redirect input from a file, or nested files containing sequences of often repeated commands. As was mentioned above, **ui** controls the interface to the IWS. It's from **ui** that the user initialized the IWS, putting it into a default state ready for data acquisition, analysis, and display. Following initialization, **ui** was used to customize the IWS configuration for the particular experiment. Most of the global system parameters were defined through **ui** commands, such as: enabling the front-end data acquisition hardware, defining which analysis algorithms to use, selection of a plot configuration for graphics display, and turning on the archiving of data.

Data Acquisition

The control of the front-end hardware was performed by a devoted program called **isdl**. **ui** passes commands to **isdl** by what in UNIX is known as a *pipe*. A *pipe* is an interprocess communication mechanism used to send data from one program to another. Its one method of accomplishing one-way communication between programs.

After **isdl** was done executing a command it read the *pipe*. If there are no commands in the *pipe*, **isdl** suspends execution until there is (referred to as read with wait or blocked read). However if **isdl** was performing a task, such as continuously acquiring data, then the read on the *pipe* would not wait (unblocked) for new commands to arrive, but would proceed with acquiring the next data scan.

While in the continuous scan mode of operation, **isdl** would execute the incoming command before processing the next scan. **isdl** did not initiate each scan for the HP 3852's unless the single scan mode was in effect. That function was done directly on one of the HP 3852's, using its internal real-time clock to initiate a data scan, and at the same time send an external trigger to the other HP 3852's, synchronizing all the data acquisition. This technique of placing the burden of timing on the front-end equipment, assured better uniformity of the scan interval, placed less demand on the CPU, and resulted in better over all user response.

After collecting all of the data for a scan, the HP 3852's performed any additional processing (such as averaging), before writing the data into its output buffer. The HP 3852 then configured itself for the next scan and waited for the time interval to expire, or for a trigger in the case of a slave unit. *isdl* in the mean time was waiting to read the HP 3852's data, after which *isdl* applied the appropriate engineering units conversions, and wrote it to *shared memory* for access by other programs.

Analysis

Before *isdl* returns to read the *pipe*, it sends a *signal* to the analysis program *rvesp* telling it to start. A *signal* is also a UNIX form of interprocess communication that is efficient at interrupting a program as a result of an external event. *rvesp* functioned similar to a spreadsheet-type code allowing the calculation of derived data.

Most of *rvesp*'s mathematics and syntax was geared towards linear algebra and matrix operations; however scalar operations were also supported. *rvesp* had a powerful data manipulation function similar to an if-condition statement processor. *rvesp* contained a well rounded tool-kit for data processing including: macro files that had argument passing capabilities, and a variety of data base functions for manipulating the data matrix.

The commands to *rvesp* were first read in through *rvisp* the reverse polish command interpreter initializer program. *rvisp* functioned similar to a compiler. Commands were parsed and interpreted, expressions were converted to reverse-polish notation, and then everything was written to a large command buffer in *shared memory*. When *rvesp* was executed, it read all of the preprocessed (compiled) operations directly from *shared memory*. Because there was no disk access (as in the batch operation of the post-run version of the code), very good real-time performance was obtained from *rvesp*.

Graphics

After the analysis and creation of derived data, *rvesp* sent a *signal* to the plotting program Quick Look DRaW (*qldrw*). *qldrw* is similar to the *rvesp/rvisp* pair in that all of the user's plot selections are predefined and loaded into *shared memory* by a user interface routine called PLOt SElect (*plsel*). Plots were the primary end product of IWS during the run.

Post-run plots of all the data over the entire run duration were provided in a run report. Plots were of such importance both in real-time and post-run that the plot configuration interface contained a great deal of flexibility. Plots were guaranteed to display the last 1024 points collected, assuming that many points have already been collected. The most basic of the options was the selection of the dependent and independent variables to be plotted. If multiple variables were to be displayed upon the axes (superimposed), the default option was to scale the axes to fit all of the data being plotted. Automatic axes scaling could be overridden, if the user desired.

qldrw allowed up to a maximum of eight variables to be superimposed upon a single plot. By default all of the plots had time as the x-axis. Time was represented as a decimal hours relative to the start of the experiment. As new data was added to the plot the display took on the effect of a strip-chart recorder. *plsel* was interactive, allowing the user to dynamically reconfigure the plots during the experiment. If the experimenter was interested in a particular signal or calculated value, then that information could be easily brought up on the screen.

An overview of the interaction between the software processes of the IWS are shown on Figure 2.

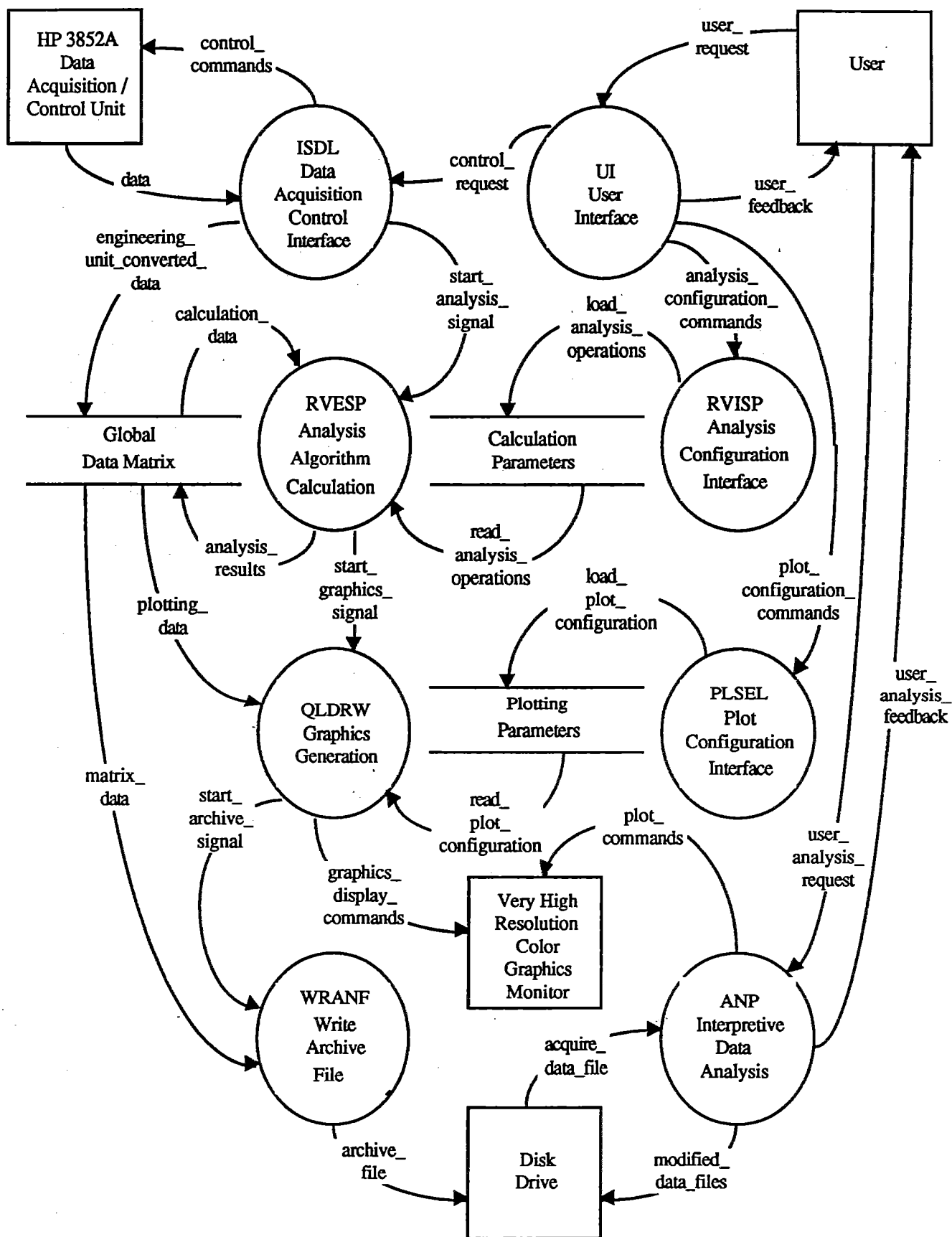


Fig. 2 Integrated Workstation Data Flow Diagram (Context Level)

Conclusion

By requiring all data entered into the IWS to be analog (0 to 10 volts) in nature, the flexibility and ease of adding new signals was almost as simple as plugging into a patch panel. Division of responsibilities between the providers of the analog signals and the IWS was mediated at the patch panel. Pre-run end-to-end checkout and test duties could be carried out independently. A disadvantage of this approach was if derived data was required from another workstation, this information must be converted back into analog signals for reacquisition by the IWS. In the process of doing this the calibration information was lost, along with possible significant figures in the derived 32-bit real number to 13-bit voltage conversion. This function was usually performed by hardware on the workstation, hence didn't steal cpu cycles, it did use up expansion slots. The HP 3852 easily accommodated other types of signals (temperatures, pulse counts) by adding the appropriate module.

The speed of the acquisition and display of data provided an accurate indication of the current performance and condition of the experiment. Powerful numeric processing available from the analysis spreadsheet allowed the physicists to observe the results of the experimental models applied to real data in near real-time. Knowing the current operating conditions of the experiment facilitated experiment scheduling decisions, assisted in the optimization of the process, and the development of algorithms. All of the data collected and calculated was archived and accessible to the experimenters for post-run analysis, modification, or report generation.

The AVLIS Program is scheduled for a major integrated enrichment demonstration in the early 1990s. The Full-Scale Demonstration Facility will fine-tune the scientific and economic concepts of the process to facilitate the transition to commercial use. Planned future work in support of this effort include: 1) moving more towards networked based data collection and distribution; 2) providing X-window user interfaces, and; 3) supporting distributed graphics for both the real-time trending and for historical data display.

This work was performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under Contract No. W-7405-ENG-48.